**Modules**

Code in Visual Basic is stored in the form of modules. The three kind of modules are Form Modules, Standard Modules and Class Modules. A simple application may contain a single Form, and the code resides in that Form module itself. As the application grows, additional Forms are added and there may be a common code to be executed in several Forms. To avoid the duplication of code, a separate module containing a procedure is created that implements the common code. This is a standard Module.

Class module (.CLS filename extension) are the foundation of the object oriented programming in Visual Basic. New objects can be created by writing code in class modules. Each module can contain:

**Declarations :** May include constant, type, variable and DLL procedure declarations.

**Procedures :** A sub function, or property procedure that contain pieces of code that can be executed as a unit.

These are the rules to follow when naming elements in VB - variables, constants, controls, procedures, and so on:

* A name must begin with a letter.
* May be as much as 255 characters long (but don't forget that somebody has to type the stuff!).
* Must not contain a space or an embedded period or type-declaration characters used to specify a data type; these are ! # % $ & @
* Must not be a reserved word (that is part of the code, like Option, for example)
* The dash, although legal, should be avoided because it may be confused with the minus sign. Instead of First-name use First\_name or FirstName.

**Data types in Visual Basic 6**

By default Visual Basic variables are of variant data types. The variant data type can store numeric, date/time or string data. When a variable is declared, a data type is supplied for it that determines the kind of data they can store. The fundamental data types in Visual Basic including variant are integer, long, single, double, string, currency, byte and boolean. Visual Basic supports a vast array of data types. Each data type has limits to the kind of information and the minimum and maximum values it can hold. In addition, some types can interchange with some other types. A list of Visual Basic's simple data types are given below.

**1. Numeric**

|  |  |
| --- | --- |
| **Byte** | Store integer values in the range of 0 - 255 |
| **Integer** | Store integer values in the range of (-32,768) - (+ 32,767) |
| **Long** | Store integer values in the range of (- 2,147,483,468) - (+ 2,147,483,468) |
| **Single** | Store floating point value in the range of (-3.4x10-38) - (+ 3.4x1038) |
| **Double** | Store large floating value which exceeding the single data type value |
| **Currency** | store monetary values. It supports 4 digits to the right of decimal point and 15 digits to the left |

**2. String**

Use to store alphanumeric values. A variable length string can store approximately 4 billion characters

**3. Date**

Use to store date and time values. A variable declared as date type can store both date and time values and it can store date values 01/01/0100 up to 12/31/9999

**4. Boolean**

Boolean data types hold either a true or false value. These are not stored as numeric values and cannot be used as such. Values are internally stored as -1 (True) and 0 (False) and any non-zero value is considered as true.

**5. Variant**

Stores any type of data and is the default Visual Basic data type. In Visual Basic if we declare a variable without any data type by default the data type is assigned as default.

**Operators in Visual Basic**

**Arithmetical Operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operators** | **Description** | **Example** | **Result** |
| + | Add | 5+5 | 10 |
| - | Substract | 10-5 | 5 |
| / | Divide | 25/5 | 5 |
| \ | Integer Division | 20\3 | 6 |
| \* | Multiply | 5\*4 | 20 |
| ^ | Exponent (power of) | 3^3 | 27 |
| Mod | Remainder of division | 20 Mod 6 | 2 |
| & | String concatenation | "George"&" "&"Bush" | "George Bush" |

**Relational Operators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operators** | **Description** | **Example** | **Result** |
| > | Greater than | 10>8 | True |
| < | Less than | 10<8 | False |
| >= | Greater than or equal to | 20>=10 | True |
| <= | Less than or equal to | 10<=20 | True |
| <> | Not Equal to | 5<>4 | True |
| = | Equal to | 5=7 | False |

**Logical Operators**

|  |  |
| --- | --- |
| **Operators** | **Description** |
| OR | Operation will be true if either of the operands is true |
| AND | Operation will be true only if both the operands are true |